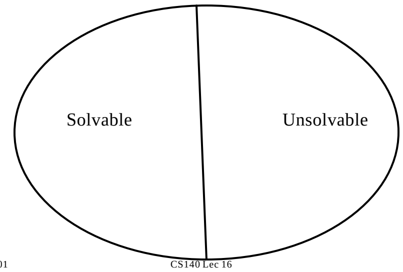
CS140: Algorithms

Z Sweedyk
Lecture 16

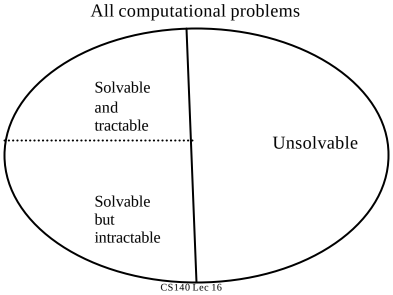
The world as we know it …

All computational problems



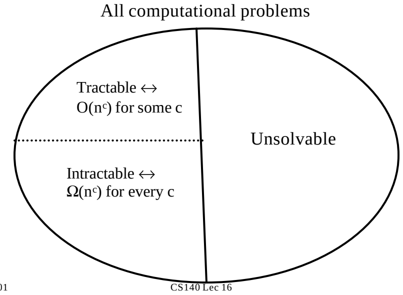
The world as we know it …

All computational problems



The world as we know it …

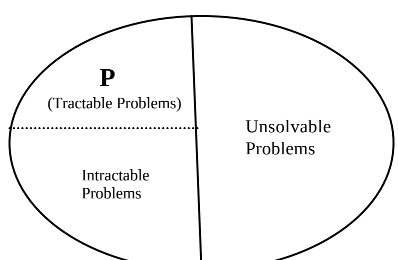
All computational problems



Why polynomial?

- Someone said so…
- It makes a lot of sense…

The world as we know it …



What's in P?

Every decision problem that has a polynomial-time algorithm: e.g.

- Is the list S of integers sorted in ascending order?
- Is the graph G connected?
- Does graph G have a MST with cost K or less?
- Does tree T have a vertex cover of size K or less?

What about Search problems?

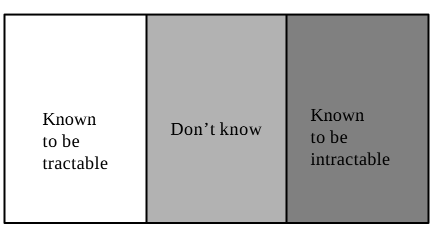
- We'll come back to that.

What is not in P?

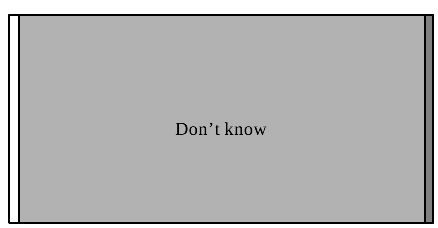
- Recognizing true statements in Presburger arithmetic
- The circularity problem for attribute grammars
- Inequivalence for regular expressions with squaring
- And others

Huh?

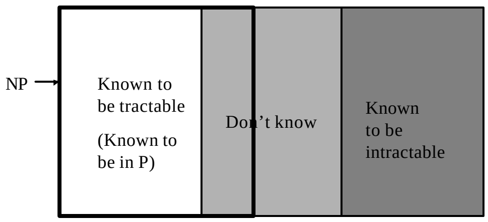
The world of solvable problems…



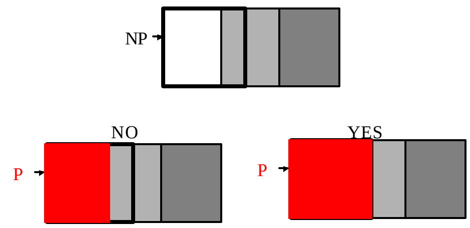
The world of solvable problems…as it seems



NP: all tractable + some don't know



Is P=NP?



About NP

- P is in NP
- Some of NP is in don't know
- NP as a class has some nice properties
- NP is the smallest class containing some don't knows that has these properties

What is NP?

- NP is the class of decision problems that have polynomial-time verifiable proofs.
- HUH?

Formal Language Theory

- A language over an alphabet Σ is a subset of Σ^*
- Examples for $\Sigma=\{0,1\}$
  - $\{01,0101,010101,\ldots\}$
  - $\{0, 11, 110, 1001, \ldots\}$
  - ϕ
  - Σ^*

Some classes of languages

- Regular
- Context-free
- Recursive
- Recursive-enumerable

Language classes

- Language classes are typically defined by the computational power needed to answer membership queries:

  Is x in L?

Some classes of languages

- Regular – Finite Automata
- Context-free – Pushdown Automata
- Recursive – Turing machine
- Recursive-enumerable – Turing machines can answer yes but not necessarily no.

Turing machine

- A simple model of a computer:
  - Finite state machine
  - R/W tape
  - Can be programmed to follow simple rules

Church-Turing thesis

- Any physically-realizable computing device can be modeled with at most polynomial-time blowup by a randomized Turing machine.

Note

- Church-Turing thesis may be disproved by quantum computers if they are found to be
  - Physically realizable
  - Provably more powerful than traditional Turing machines

More classes

- Membership questions can be answered by resource-bounded Turing machines
  - Limit time
  - Limit space
  - Limit randomness

P

- Membership questions can be answered by resource-bounded Turing machines
  - Limit time – polynomial
  - Limit space
  - Limit randomness – none

More classes

- Membership questions can be answered by resource-bounded **non-deterministic** Turing machines
  - Limit time
  - Limit space
  - Limit randomness

NP

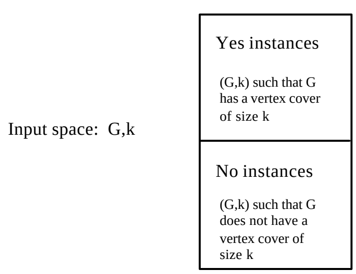
- Membership questions can be answered by resource-bounded **non-deterministic** Turing machines
  - Limit time - polynomial
  - Limit space
  - Limit randomness - none

Decision Problems

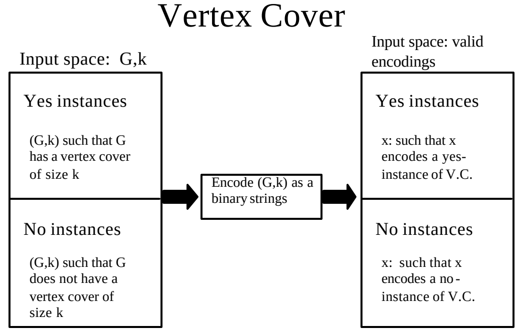
- Computational problems in which the output is Yes or No.

- Decision problems can be posed as membership queries.

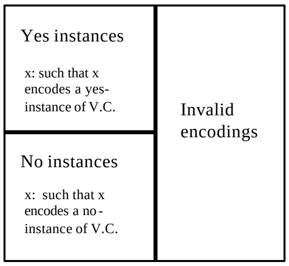
Vertex Cover



Vertex Cover



Language over $\{0,1\}^*$



Encoding Rule

- It is easy to determine whether or not a binary string is a valid encoding.
- A problem of size n can be encoded in poly(n) bits.

⬇

- Notion of tractability is preserved.

Our outlook

- Natural problems
  – Ignore coding issue unless it matters
- Decision version
  – What about search?
- Objective to distinguish tractable from intractable

Shopping Bag Problem

n items of weight at most B

- Natural problems
  – Ignore coding issue unless it matters

Shopping Bag Problem

n items of weight at most B

- Natural problems
  – Ignore coding issue unless it matters
  – Coding clarifies what we mean by “input size”

Shopping Bag Problem

n items of weight at most B

- Natural problems
    - Input size is n lg(B) bits

Shopping Bag Problem

n items of weight at most B

- Natural problems
  - Input size is n lg(B) bits
- Objective: An O(nB) algorithm does not prove tractability

Our outlook

- Decision version
  - What about search?
    - If the decision problem is intractable then the search problem is intractable. Why?

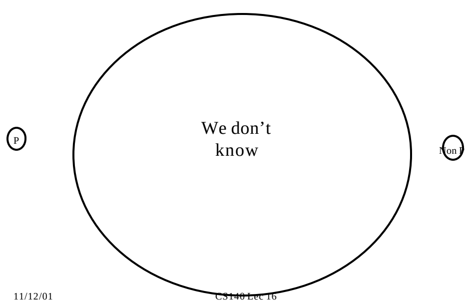
Our outlook

- Decision version
  - What about search?
    - If the decision problem is intractable then the search problem is intractable. Why?
    - Typically, if the decision problem is tractable then so is the search problem. Why?

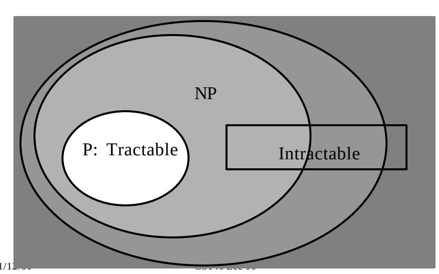
Our outlook

- Objective to distinguish tractable from intractable
  - However meager this may be

The world as it seems…



The world as we believe it to be…



NP

- Languages that can be posed as
  {x | ∃y such that P(x,y)}

  where P(x,y) is checkable in time poly(|x|)

Example

- VC:
  - x=(G,k)
  - y is a vertex cover of G containing k or fewer vertices
- 3-coloring:
  - x=G
  - y is a function mapping V to {red,blue,green}

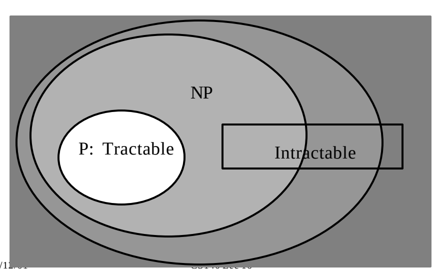
NP

- Languages that can be posed as
  {x | ∃y such that P(x,y)}
  where P(x,y) is checkable in time poly(|x|)
- y is called a witness (or proof) for x

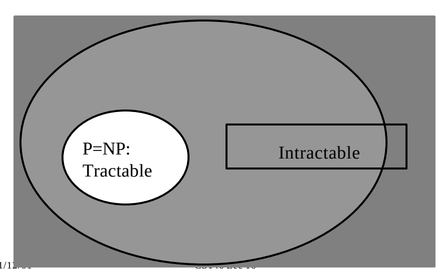
NP: Other characterizations

- Languages decidable in polynomial-time by a non-deterministic Turing machine
- Languages that have probabilistically checkable proofs using a constant number of queries and logarithmic randomness

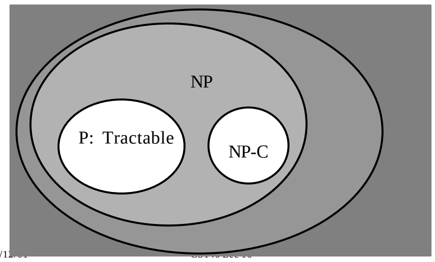
The world as we believe it to be…



But what could be…



The world as we believe it to be…



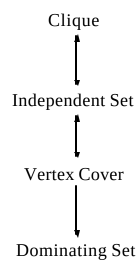
NP-complete

- If A is NP and B is NP-complete then
  $A \propto_p B$
- If any NP-complete problem is tractable then every NP problem is tractable.

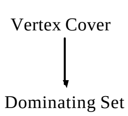
NP

- Is it in NP?
  - Is it also in P?
  - Is it NP-Complete?
  - Else?

NP-Completeness Map



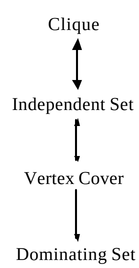
NP-Completeness Map Legend



VC $\propto_p$ DS:

1. If VC is NP-hard then so is DS.
2. If DS can be solved efficiently then so can VC.

NP-Completeness Map



Clique ↔ Independent Set

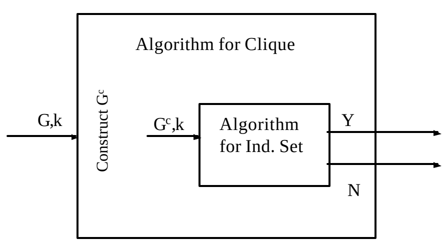
For G = (V,E) the complement of G is

$G^c = (V, V \times V - E)$

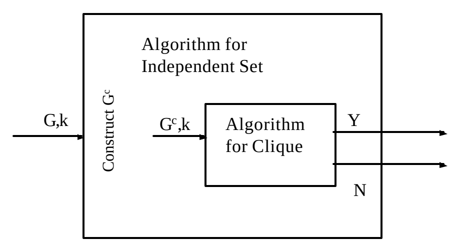
Clique ↔ Independent Set

- A clique in G is an independent set in G^c.
- A clique in G^c is an independent set in G.

Clique $\propto_p$ Independent Set



Independent Set $\propto_p$ Clique



T_{clique} (n,m)

$T_{clique}(n,m) = n^2 + T_{ind\text{-}set}(n,n^2\text{-}m)$

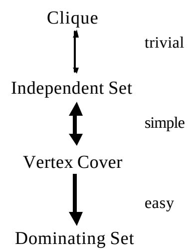
Reduction is poly(n,m)

T_{clique} (n,m)

$T_{clique}(n,m) = n^2 + T_{ind\text{-}set}(n,n^2\text{-}m)$

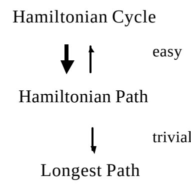
If $T_{ind\text{-}set}$ is polynomially-bounded then so is T_{clique}.

NP-Completeness Map



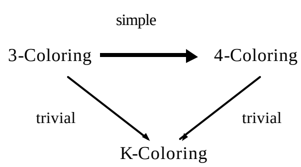
11/12/01 CS140 Lec 16 61

NP-Completeness Map



11/12/01 CS140 Lec 16 62

NP-Completeness Map



11/12/01 CS140 Lec 16 63